ECOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE
(EPFL)

MASTER THESIS

3D hand reconstruction from RGB-D/RGB video frames in real-time

---

Author:
Aleix TODA MAS

Supervisor:
Dr. Alexandre ALAHI

Visual Intelligence for Transportation VITA
School of Computer and Communication Sciences

April 30, 2019

"*Don't be misled by the name 'artificial intelligence' - there is nothing artificial about it. AI is made by humans, intended to behave by humans, and, ultimately, to impact humans' lives and human society.*"

Fei-Fei Li

ECOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE (EPFL)

Abstract

Faculty Name

School of Computer and Communication Sciences

Master of Artificial Intelligence (MAI)

3D hand reconstruction from RGB-D/RGB video frames in real-time

by Aleix TODA MAS

We present a pipeline able to extract 3D real-world measurements from RGB-D images with high accuracy in real-time. In order to evaluate the method, a dataset with the ring finger width real measure of all subjects has been recorded with a depth camera, and we have used our ring finger width predictions to estimate the ring size of the user. Due the nonexistence of other ring size estimators, we present qualitative and quantitative results based only on our method. One of the main steps of our pipeline is the hand pose prediction, therefore, with the aim of providing an alternative of that could manage the hand pose estimation without the need of depth data, we have extended a 2D multi-person body pose estimator to be able to predict the 2D body-hand/hand pose, which is, to our knowledge, the first public 2D bottom-up approach for multi-person, multi-hand pose prediction. Moreover, we present two novel training strategies, privileged masking and data fusion. The first exploit the case in which two datasets with the same images but mutual exclusive annotations are available, but because any of them has all the target instances annotated, the privileged masking will avoid the loss penalization of the network in case a prediction of an instance without ground truth annotations is really there (e.g. if there is the same dataset obtained by two different sources and in one there are humans without annotations in the images, but the humans that are annotated have hands and body ground truth, and the second dataset has all bodies in all images annotated but any hand ground truth). The second strategy allow to train our network using mixed images of datasets with different images but with only some common parts of the target instances annotated (e.g. when a dataset has only the right hands annotated and another has both hands annotated). Experiments with some of the most popular 2D hand pose datasets have been carried, and the results demonstrate the high performance of our new method.

Acknowledgements

I would like to thank Alexander Alahi and Sven Kreiss for their support and guidance in this project. But specially I would like to thank my family, who even when I took directions far from what they always have suggested, I always have had their support.

Contents

Acknowledgements v

1 Introduction 1

2 Related work 3

3 Ring finger width dataset 5

4 Method 7
- 4.1 Data acquisition . . . 7
- 4.2 Hand detection . . . 7
- 4.3 Outliers removal . . . 8
- 4.4 Depth based hand pose prediction . . . 8
  - 4.4.1 Preprocessing . . . 9
  - 4.4.2 3D Joints position prediction . . . 9
- 4.5 Ring finger width prediction . . . 10
- 4.6 Final prediction . . . 10

5 RGB-based hand pose detection 13
- 5.1 OpenPifPaf4hands . . . 13
- 5.2 Datasets . . . 13
- 5.3 Instance models . . . 14
- 5.4 Training . . . 14
- 5.5 Decoder parameters tuning . . . 16
- 5.6 OpenPifPaf4hands evaluation . . . 17

6 Experiments 25

7 Conclusions 29

8 Future work 31

A Finger width measurement 33

B OpenPifPaf4hands 35
- B.1 Training loss visualization . . . 35
- B.2 Prediction tuning: decoder parameters setting . . . 36

Bibliography 37

List of Figures

3.1 Red: ring finger width at the ring location. Blue: ring finger width at the dip joint. 5

4.1 Images of each of the steps in the algorithm described in the subsection 4.5. In the top left, the point cloud overlapped with the hand keypoints. In the top right image, the 2 concentric circles of the region of interest could be seen. In the bottom left image, the target region has grown until reaching all the points of the ring finger inside the region of interest (points in red), the points in green are the ones that will not be included in the target region due to its distance from the growing region. In the bottom right image, the orientation of the target region using it's eigenvalues could be seen, while the finger width measurement will be the average distance between the green and yellow points. 12

5.1 Left images: privileged masking for the hand model. Right images: privileged masking for the Hand-body-nzl model. Top row: available keypoints of the original MPII dataset. Middle row: available keypoints of the MPII+NZL dataset. Bottom row: black bounding boxes representing the masked pixels around the instances without annotations. 15

5.2 Visualization of the paf vectors during inference. In the left, the output paf vector fields for the left elbow with an image which rescaling at the preprocessing stage provided retained too much resolution. In the right, the output paf vector fields for the wrist with a proper rescaling at the preprocessing stage, where it could be clearly seen that only in the left wrist all paf vectors point to the same point. 16

5.3 OpenPifPaf4hands predictions of the MPII dataset using the hand model. . . . 17

5.4 OpenPifPaf4hands predictions of the NZL dataset using the hand-body-nzl model. 18

5.5 OpenPifPaf4hands predictions of the Panoptic dataset using the hand model. With a red frame, an unsatisfactory estimated hand pose is shown, in which only the palm keypoint is close to the ground truth. It should be noted that due that on the Panoptic dataset only the annotations of the right hands are provided, we only predict the hand pose of the right hands. 18

5.6 PCK plots with distance normalized for MPII+NZL dataset, only the MPII and only the NZL subdatasets. Only the images where at least one keypoint has been predicted has been included in the plot data. 21

5.7 PCK plots with the error in pixels distance from the ground truth to the predicted keypoints for MPII+NZL dataset, only the MPII and only the NZL subdatasets. Only the images where at least one keypoint has been predicted has been included in the plot data. 22

5.8 PCK plots with distance normalized and without normalization for the Panoptic dataset. 23

5.9 MPII+NZL dataset. Orange: total number of images in the dataset. Blue: total number of images without any keypoint predicted. 23
5.10 Panoptic dataset. Orange: total number of images in the dataset. Blue: total number of images without any keypoint predicted. 24
5.11 Prediction times for the MPII+NZL dataset and it's subdatasets. 24
5.12 Prediction time for the Panoptic dataset. 24

6.1 Examples of positive and negative finger-width predictions. 27

A.1 Capture of the graphical interface of the finger measurement working in real-time. The window above show the point cloud, in the bottom left window the RGB image extracted from the point cloud and the bounding box estimated by the hand detector could be seen. In the bottom right window, the steps of the finger measurement explained in section 4.5 could be seen. 33

B.1 Target and prediction at epoch 0 of training for pif intensity of the left hand palm. 35
B.2 Target and prediction at epoch 150 of training for pif intensity of the left hand palm. 35
B.3 Target and prediction at epoch 0 of training for paf vectors pointing to the left hand palm. 36
B.4 Target and prediction at epoch 90 of training for paf vectors pointing to the left hand palm. 36
B.5 Predictions for the hand-body-nzl model with different threshold values in MPII dataset crowded images. 36

List of Tables

5.1 Threshold values for the different models and datasets. 16

6.1 Average errors in mm of the finger width predictions for each hand of each of the 14 subjects of our dataset. The left half present the results with relaxed constraints and the right half the results with strict constraints The presence columns indicate the number of instances of that hand and subject have been considered valid for the measurement. In the bottom 4 rows, the maximum, minimum, average and sum of all the errors for each column are presented. . 26

Chapter 1

Introduction

Real scale three dimensional instance reconstruction is a key step in computer vision. With the advent of the affordable depth cameras and the advance of the state of the art of the hardware like the GPUs, computationally expensive and memory demanding models like the required to predict the 3D mesh of an instance or a local map could be used even in real-time. The applications of 3D reconstruction go from virtual or augmented reality (gaming, educational apps, clothes size prediction for online shopping, etc) to human-robot interaction, to name a few. The Microsoft Oculus, Google Daydream or Magic Leap One, and even the last smartphones, are only some of the examples of the current devices that use 3D reconstruction for applications that go from virtual or augmented reality (gaming, educational apps, clothes size prediction for online shopping, etc), face recognition to re-identify an user, personalized avatar creation to human-robot interaction, to name a few.

Inside the huge field of 3D reconstruction, this project will be focused on the hands. Hands are a fundamental tool for humans. We not only use them to communicate with others but also as the main way to manipulate our environment. Therefore, to understand their movements in a precise manner is crucial if we want the devices with artificial intelligence to understand us properly and therefore, to behave as expected, for instance, if a robot has to understand sign language or an autonomous vehicle if a human makes a stop sign).

Hand pose prediction is a simple way to represent the hand and nowadays is the most used strategy to detect and track the hands position and it's movements. Pose prediction consists of the extraction of the location of hand keypoints, that are then used to build the skeleton of the hand. In computer vision, this could be done by means of RGB, RGB-Depth or just depth images,

Hand pose is usually more challenging than their related most common body pose approach due to the reduced size of the hands, the additional degrees of freedom, the occlusion (with objects, with the hand itself or with the other hand) and the fact that hands can be moved at more velocity than the rest of the body, provoking that sometimes they appear blurry in the images.

The methods that tackle hand pose prediction could be divided in 3 groups:

1. **Model-based**: A predefined hand model is aligned with the input to try to generate the most accurate hand that satisfy the pose priors.
2. **Discriminative**: The location of the hand keypoints are directly predicted, usually using a deep neural network (DNN). Discriminative methods outperform model-based methods in means of accuracy but sometimes the predicted poses have implausible shapes.

3. **Hybrid**: At the first stage behave like a discriminative method and once the keypoint locations are predicted, a hand model is used to discard unsatisfactory shapes.

If the scale of the target instance is not needed, 2D single images could be used to extract the location of the keypoints in relation to a root keypoint, used as a reference. Even though, in some cases the scale is a relevant parameter. Scale prediction from single RGB images is indeed an ill-posted problem due to the ambiguity. To address this issue, the following approaches could be used:

1. **Reference object**: If an object of known size is present in the image, the pixel/distance relation could be computed and then the scale of the rest of the instances in the image could be extracted. The need of a reference object and the inaccuracies produced by the rotational and depth difference between the reference and target planes makes this approach undesirable in most of the cases.
2. **Inertial Measurement Unit (IMU)**: Making use of an RGB camera with an IMU, if enough keypoints are present in consecutive images, the data provided by the sensors can be used to compute the camera registration in each image. This algorithms can run as fast as 100 Hz and are widely used in robotics.
3. **Visual Simultaneous localization and mapping (SLAM)**: Making use of keypoints matches between consecutive frames the camera registration could be computed. Can be used independently or fused with an IMU algorithm, tough, SLAM is works only at 20 Hz and is only used to provide error corrections.
4. **Depth cameras**: Despite the wide range of different ways that this cameras use to compute the depth, the common target is to estimate the distance from the camera to each pixel in the image. Methods based on depth cameras are becoming the state of the art in 3D reconstruction and 3D pose prediction.
5. **Deep Neural Networks (DNNs)**: With the advent of larger RGB/RGB-D datasets with 2D/3D annotations, some methods like [17] achieved to extract the 3D hand pose directly from RGB images.

Due to the difficulties on 3D reconstruction evaluation, in this project the measurement of the width of the ring finger will be used to quantitatively evaluate our model. Hence, the ring finger width measurement could be directly translated to the most suitable ring size for an user. This specific application has never been addressed publicly previously and there is no an easy way to evaluate the obtained results. To tackle this issue a new dataset with the ring finger widths and ring size of all users has been recorded. Despite the reduced dimensions of the dataset, it will allow not only to evaluate the algorithm qualitatively but also quantitatively. Moreover, our algorithm not only can work offline but also is able to make predictions in real-time.

As a parallel work, we wanted to explore alternatives to manage the hand pose prediction from RGB images, avoiding the necessity of the depth data. Hence, a multi-person 2D body pose predictor has been adapted to predict the 2D hand pose. This new method is, to our knowledge, the first 2D bottom-up multi-person multi-hand pose predictor, which can agglomerate the detection and pose estimation of all the hands in an image into one single step.

Chapter 2

Related work

The ambiguity of the scale in 3D reconstructions from single RGB images was addressed in [18] with the previous mentioned approach with a camera with an IMU. The results show that with enough matching keypoints in consecutive frames the scale estimation error was below 1-2%.

A wide study of the hand pose from depth images current challenges is studied in [13], where 3D CNNs show their superior ability to capture the input depth information at the expenses of needing more computation time and memory than 2D CNNs. Detection-based methods, producing a probability density map for each joint, perform better than regression-based methods, which directly estimate the location of each keypoint in the image or the angle between keypoints. Hierarchical methods are more effective when there is occlusion because of it's capacity to grow the hand pose from small local regions.

In [7], a pipeline of 3 steps is used to estimate the hand pose from single RGB images. The method starts by cropping a bounding box for each detected hand in the image, then, the 2D hand pose is predicted with the algorithm of [10] and finally, a post-processing step fits the predicted 2D hand pose into a 3D hand model.

The work of [19], claims that the misalignment between the 2D prediction and the hand model was the most important issue of the approaches that use an intermediate 2D pose prediction step. In contrast, they set the center of mass (COM) of the hand as the reference point, then the outliers produced by the acquisition noise are removed and the missing data is reconstructed before being provided to a 3D convolutional neural network that directly outputs the 3D locations of the hand keypoints relative to the COM. Therefore, any post-processing step is avoided.

In many works [20], [17], [21], [22], [19], synthetic data was used for data augmentation, addressing the common issue of the reduced number of hand pose datasets, the small number of images and the lack of variability of subjects, hand poses or backgrounds. The results demonstrate that using synthetic data alone or mixed with real data could provide a boost to the inference accuracy.

The main reason behind the lack of large hand pose datasets, is the tedious and time consuming work of manually annotate the keypoints or the necessity to use special markers, which decrease the quality of the dataset. In [10], a method to automatically annotate instances is built, using a room with 31 fixed cameras with known position they take RGB recordings from all at the same time. Then, a hand detector is used to crop an image of each hand and the hand pose is estimated from each viewpoint. Hence, the RANSAC algorithm is used to match the extracted hand keypoints from all the images, and after rejecting the predictions too different from the rest, the average of all the predictions is taken as the result of that iteration. Then, the hand pose detector is trained with the images from all the viewpoints that produced non-rejected poses. This process is repeated iteratively until the satisfactory accuracy is reached.

The work presented in OpenPose [8] is probably the most used method to predict the full body pose. Their work consist on a bottom-up approach that builds the connections of all the body skeleton from Part Affinity Fields, they achieve to predict the body, face, hands and foot in real time. Their strategy is to first find the body keypoints to follow-up cropping subimages of each individual hand using the predicted wrist and elbow keypoints. Then, the method of [10] is used to predict the hand pose of each hand separately.

Chapter 3

Ring finger width dataset

Due to the non-existence of public RGB-D datasets with finger measurements ground truth, a new dataset has been recorded to evaluate our method. The dataset consists on the measurement of the dip joint width of the ring finger and the width at the ring place of the ring finger with a caliper. In the figure 3.1), the two measurement locations could be seen. Additionally, the ring size of each user was also taken using a set of rings of all available ring sizes.

FIGURE 3.1: Red: ring finger width at the ring location. Blue: ring finger width at the dip joint.

For each of the 14 subjects in the dataset (13 males and 1 female), both hands were separately recorded first, with the hand supported in a surface and then with the hand separated 30 cm from any other object. During each recording, the depth camera was moved in a circular movement around the hand, assuring a viewpoint span of 180 degrees. An average of 628 frames were captured from each subject, making a total of 8793 images with RGB-D data.

Chapter 4

Method

Our method tackle the finger sizing through a pipeline of 6 steps. In the following subsections, each step is described.

4.1 Data acquisition

The Point Cloud Library (PCL), written in C++, has been used to manage the acquisition of the 3D point clouds provided by the depth camera with the maximum velocity and robustness.

The depth camera used in this project is the ASUS Xtion Pro [23], able to take 640x480 RGB-D images at 30fps. The Xtion is a structured light camera, meaning that a mesh of IR light dots with a known pattern is projected and then the difference between the projected pattern and the one that the camera captures, deformed by the irregularities of the surfaces, is used to compute the distance of each pixel from the camera (depth). Structured light cameras provide good accuracy avoiding the heavy computational demand of the stereo cameras. One pitfall is that due the IR nature of the projected pattern, the sunlight can interfere the readability of the projected pattern by the camera.

The data provided by the camera is a point cloud containing 4 data fields for each pixel, which could be expressed as $p_{x,y} = \{r, g, b, d\}_{x,y}$, referring as the red, green, blue and depth of the pixel located in the x, y coordinates in the image plane. The advantage of the point cloud representation is that the 3D visualization could be easily extracted from the x and y coordinates plus the depth of each pixel. In some literature, this point cloud representation from single image data is also named 2.5 dimensional data, because the occlusion makes the 3D reconstruction incomplete.

4.2 Hand detection

Once the point cloud is obtained, the RGB image without the depth is passed to a hand detector. This step is common in most of the hand pose detection pipelines, like in the work of [7], in which the hand detector of the work of Convolutional Pose Machines [11] is used first, following the hand pose estimator of the same CMU-PerceptualComputing Lab [10] is in charge of 2D hand pose prediction from the RGB cropped image and finally, a 3D hand model is used to obtain the 3D joint locations from the 2D predictions.

A learning single shot detector SSD [4] coded in tensorflow and trained on the Egohands Dataset [5] have been used as the base for the hand detector. The original code has been adapted to the specific needs of this project, making it lighter and arranging the format of the input/output data. Due that the Egohands Dataset contain only first-person view dataset, the detector performs quite well on our target images, in which usually only the hand of the target person could be seen.

This step has the target of removing the unnecessary data from the point cloud, retaining just the points inside the bounding box predicted by the detector. For simplicity, it has been considered that only one hand could be present at the same time in the image, if more than one hand is present, just the one detected with more confidence will be processed.

A single shot detector SSD has been chosen because it provides fast results with an acceptable accuracy rate, allowing a real-time detection.

4.3 Outliers removal

If a hand presence has been detected and the remaining cloud point after the cropping of the bounding box contain at least 1024 points, the next step is the outlier removal, which consists on two sequential steps:

1. Depth pass-through filter: Depth information could be used to easily remove the background in most of the situations, specially when the hand is not close to any object or surface. This filter search for the point closest to the camera, which is supposed to be one point of the target hand, and following all the points deeper than 10 cm, meaning points farther than 10 cm than the hand from the camera viewpoint, are removed.
2. RGB region growing segmentation: In this case, not only the spatial distances but also the color distances between all the points are used to group the points in different instances. Therefore, all the instances with less than 1024 points are discarded and from the remaining ones, the one with the less color distance with a human skin reference is the selected as the hand instance. This algorithm is very powerful and has demonstrated to be able to separate efficiently the hand from the background, being more accurate than a pass-through filter when the hand is not enough spatially separated from the surrounding surfaces or objects.
3. Statistical outlier removal: This algorithm comes with the PCL library and after computing the mean and standard deviation of the distances between all the points, the points statistically considered outliers are deleted. This filter is really helpful to deal with depth acquisition noise.

4.4 Depth based hand pose prediction

Up to now the hand point cloud has been obtained, nevertheless, the exact position in which the measurement has to be done is still unknown. Therefore, a hand pose estimation algorithm will be used in order to predict the position of the key joints and proceed to perform the finger width measurement in the proper location.

To select the proper hand pose estimation algorithm, the paper of the Hands in the Million Challenge [13] was analyzed. The challenge consisted on the evaluation of the most recent developments in 3D hand pose prediction at the publication date (2018). The methods of the participants were evaluated in a wide range of poses, collected from the BigHand2.2M and First-Person Hand Action dataset (FHAD).

Finally, despite its 3rd position in the challenge, the Hand PointNet [6] had been chosen because of it's trade-off between accuracy, reduced processing time and being able to work with point clouds. This method takes an unordered 3D point cloud, which describes the surface of the hand, and after properly preprocessing it, a deep convolutional neural network is used to extract discriminative features in a hierarchical way. The hierarchical approach used in this method work in 3 different stages, also named abstraction layers. In each stage, one centroid is selected for each local region to following use an individual PointNet to extract features of the k-nearest neighbors of each centroid, then the extracted features are combined and passed to the next level. In the last level, an unique PointNet extract a feature vector from all the remaining features. In section **??**, additional information about the training of the Hand PointNet is provided.

4.4.1 Preprocessing

The preprocessing consists on the following steps:

Normals computation: In order to retain the maximum information from the hand surface, the computation of the normal vectors for all points in the point cloud is made before the downsampling. For each point, its 30 nearest neighbors are used to compute its normal vector. Due that the vector could be pointing perpendicularly to any of the two hand surface sides, the pointing direction is automatically corrected to point to the camera viewpoint.

Oriented Bounding Box (OBB) and data centering: This step really makes a difference when working with data with such variability as hands, which can come in many orientations, sizes, poses and occlusions. With the aim of reducing this variability, in this step the minimum bounding box containing all the point cloud is computed. Then, the eigenvalues of the point cloud are extracted and used to align the eigenvector associated to the biggest eigenvalue with the x axis and the eigenvector associated to the second biggest eigenvalue with y axis in the camera world coordinates. After this translation, either the palm or the back of the hand surface should be perpendicular to the camera viewpoint and the middle finger parallel to the x axis.

Downsampling: To reduce the computational complexity and to make the model less prone to overfitting, the point cloud is randomly downsampled until there are only 1024 remaining points.

Normalization: To reduce the size variability, the data is normalized in such way that the data in all dimensions is divided by the largest distance of the data in all dimensions, which usually become the x dimension, that at this stage represents the distance from the wrist to the fingertip of the middle finger.

Farthest point sampling: This step cluster the cloud points in local subgroups in order to prepare the data for the abstraction layers of the Hand PointNet. Cluster sizes depend on the abstraction layer level going from smaller to bigger groups. For the first layer, there will be 512 local regions, while for the second layer there will be 128 groups.

4.4.2 3D Joints position prediction

The PointNet networks extract one 128 and 256 dimensional feature vectors for each of the local regions in the first and second abstraction layers respectively. Therefore, the PointNet in third abstraction layer extracts a 1024 dimensional global feature vector which is provided to three fully-connected layers to output the final vector. This final vector still not directly contain the 3D positions of the hand keypoints, because this feature vector is represented in a latent space, with less features than $3 \times J_n$, where J_n are the number of hand keypoints to predict.

During training, an average of the PCA coefficients and PCA means used to create the OBB of all hands in the training dataset had been stored. Therefore, that PCA coefficients and PCA means are used to recover the desired 3D keypoints from the final vector.

The code of the Hand PointNet was available in the author's web [6], initially prepared to train on the MSRA hand dataset [14]. Even though, we had to code again most of the algorithm with C++ in order to assure the compatibility with the depth camera libraries, make the algorithm faster and make it more able to generalize to new hand orientations. After some trials on our ring finger width dataset and an qualitative analysis of the results, the decision of using other dataset for training the hand pose detector was taken. From the available RGB-D hand pose datasets, the NYU dataset [15] was the next option due it's greater quantity of data. Then, all the code was adapted to this new dataset and it's singularities like the different number of keypoints. The Hand PointNet trained on the NYU hand dataset provide more accurate predictions on our dataset, therefore, it was selected as the model to use for the final pipeline.

4.5 Ring finger width prediction

After computing the hand pose, both point cloud and predictions locations are translated to the real-world again. Once the keypoints positions have been obtained, the location of the ring finger, and even more specifically, the location of the ring finger dip joint could be known. Hence, an intermediate point between the dip joint and the palm was used as the center of the region of interest. The region of interest is a circular area with two concentric circles, the average distance between the points of the point cloud inside the inner circle was computed to be used as a reference. Therefore, the finger width prediction algorithm start with all points inside the inner circle and continue growing with the addition of the points inside the outer circle that are at less than $\times 1.5$ times the computed reference average distance. The algorithm keep including new points to the target region until there are no more points inside the outer circle that fulfill the distance requirement. At this stage the target region includes only points from the ring finger, having avoided the inclusion of points from other parts of the hand. Following, the eigenvalues of the target region are computed to know which direction represent the finger width and which the finger length. Finally, an average of 10 points at each side of the finger width are used to compute the estimated finger width.

4.6 Final prediction

Taking advantage of the fact that the final ring finger width don't have to be done using just one image, N measures are used to do so. The Scipy open-source Python package is used to compute the probability density function (pdf) and the skewed pdf, then an average of both is used to predict the final measure.

Because the purpose of this measurement is to provide an approximation of the proper ring size for the user, and knowing that ring sizes differ in one mm between them, if the difference between the final measurement and the truncated value of the final measurement is greater than 0.65, the predicted ring size is the truncated value of the final measurement plus 1. Otherwise the predicted ring size is the truncated value of the final measurement plus 2.

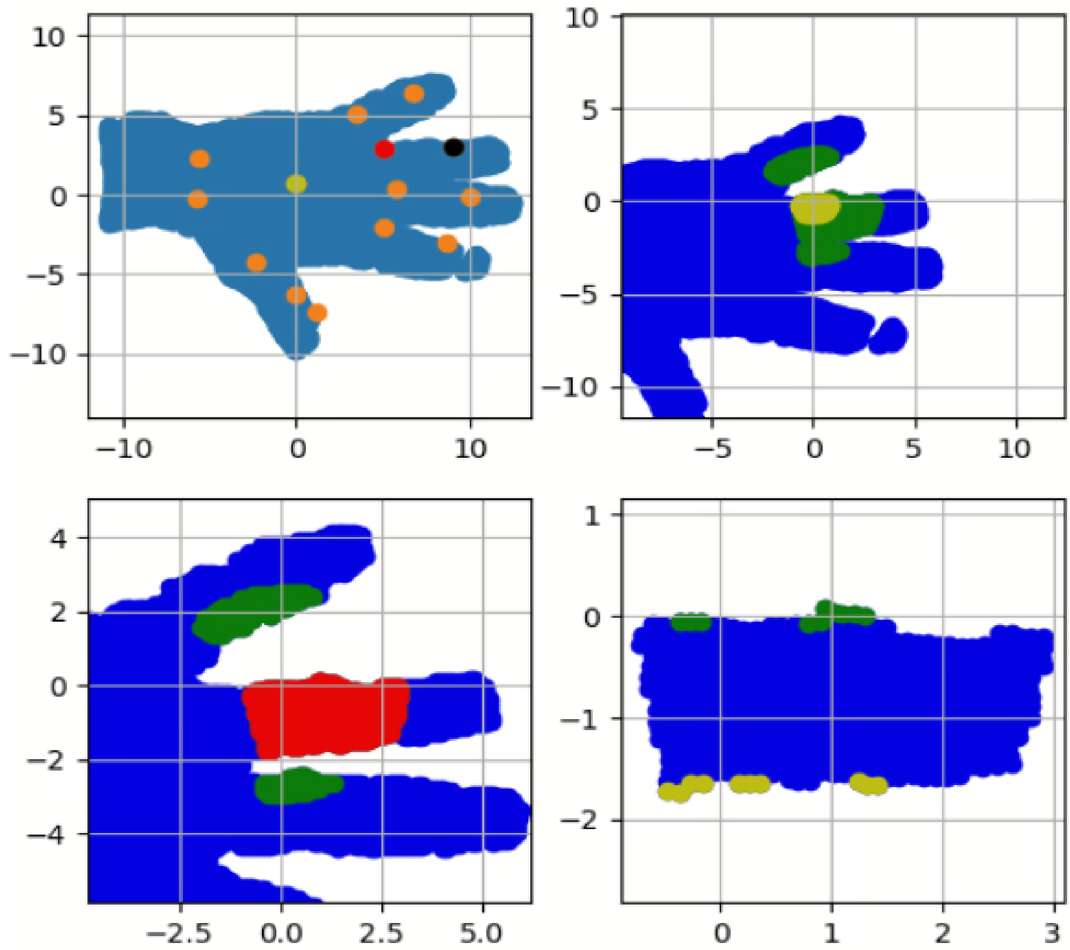

FIGURE 4.1: Images of each of the steps in the algorithm described in the subsection 4.5. In the top left, the point cloud overlapped with the hand keypoints. In the top right image, the 2 concentric circles of the region of interest could be seen. In the bottom left image, the target region has grown until reaching all the points of the ring finger inside the region of interest (points in red), the points in green are the ones that will not be included in the target region due to its distance from the growing region. In the bottom right image, the orientation of the target region using it's eigenvalues could be seen, while the finger width measurement will be the average distance between the green and yellow points.

Chapter 5

RGB-based hand pose detection

To deal with the hand pose detection, two different approaches have been studied.

5.1 OpenPifPaf4hands

As an alternative to the RGB-D based hand pose predictor that we have used in the finger measurement pipeline, we also wanted to provide another approach to deal directly with RGB images without the depth data. As mentioned in the section 2, most of the 3D hand pose predictors tackle first the 2D estimations to then predict the final 3D pose. Following the philosophy of that methods, we address first the 2D pose prediction with the adaptation of a recently published multi-person body pose predictor named OpenPifPaf [2]. OpenPifPaf is a bottom-up approach focused on the body pose prediction of the COCO dataset [25]. We adapted the method to be able to handle any instance model, with its particular keypoints and skeleton. We also adapt the resizing of the input images to our target datasets.

Despite the nonexistence of any other bottom-up approach for hands pose, OpenPose [8] catch our attention because of the use of the body pose at the first stage, and then they use the wrist and elbow keypoints to predict a bounding box for each hand. Following, the hand pose predictor of [10], from the same laboratory, is used individually for each of the obtained bounding boxes. We want to go a step further and estimate the hand keypoints at the same time as the body ones, and not only for each human independently but for all of them at the at the same time. In this way, the hand detection and the hand pose prediction of our finger width measurement will be fused in a single step.

5.2 Datasets

In order to evaluate OpenPifPaf4hands, we have used the same datasets used in [10] with the addition of the original MPII body pose dataset:

- **Original MPII [24]**: The MPII contain images extracted from several Youtube videos, providing a huge variety of environments, number of people and actions. Due that the images come from the video recording of humans in movement and in some cases the quality is poor, there are lots of blurry persons, specially their hands.

  Will be used to improve the training of our model. It contain the body annotations for all the humans in the images, without any hand annotation.

- **MPII+NZL**: This dataset consists of the mix of two different datasets in which originally only the body annotations were given. Nevertheless, the authors of [10] manually annotated some of the hands. MPII dataset has already been presented, while the second one is a video recording of different subjects telling histories in the New Zealand Sign language. In it, there is always only one subject in the image, with only the upper-body visible. For each annotated subject, it's body and visible hands keypoints are provided. The problem is that in the MPII part of this dataset not all the humans in all images are annotated. Because until now the hand pose has been always only taken in a top-down approach, facing hand by hand, this didn't represent any inconvenience. In contrast, if OpenPifPaf4hands is trained in this dataset, it learns that not all the hands should be predicted, but only some of them. Additionally, some of the annotations of the body keypoints are given in an unexpected order.

- **Panoptic**: All the images were recorded inside the same room, with different subjects performing actions, sometimes interacting with objects. Only the right hands are annotated, but in this case the annotation for all humans is given.

5.3 Instance models

We have created 2 different models to train our method:

1. **Hand model**: this model contain 21 keypoints for the hand, containing the joints of the fingers, the tips and one keypoint for the palm. Therefore, this method don't make any distinction between the left and right hands.
2. **Hand-body-nzl model**: Due the unexpected ordering of some of the body annotations of the MPII+NZL dataset and that in the NZL images, only the upper-body appeared, a model with the two hands connected by the elbows and shoulders was created. This model has the particularity that the left and right hands are treated as different particular keypoints.

5.4 Training

In order to preprocess the input data, we have used three different strategies:

- **Default**: This is strategy is basically to train with the datasets as they come, which in the case of the MPII+NZL produces the undesirable effect of teaching to our network that not all hands in the image have to be predicted, increasing the number of images without any prediction and reducing the cases in which more than one person hands are detected in an image even if there are more hands in it.
- **Privileged masking**: To avoid the negative effect of the non-annotated humans in the MPII+NZL dataset, the **privileged masking** had been created. This strategy takes advantage of the annotations of all bodies of the original MPII dataset to identify all bodies in the MPII+NZL that are not annotated. Therefore, all the bodies without annotations are masked to avoid the false loss produced when a non-annotated real hand is predicted. In the same way, when the hand model is used, not all non-annotated bodies but only the hands of the non-annotated bodies are masked. To know the place and dimensions of the bounding box necessary to mask the hands of the non-annotated bodies, an approximation using the elbow and wrist keypoints of the original MPII dataset have been used. In the figure 5.1, examples of the privileged masking for both of our keypoints models, hand and body-hand-nzl, are shown.

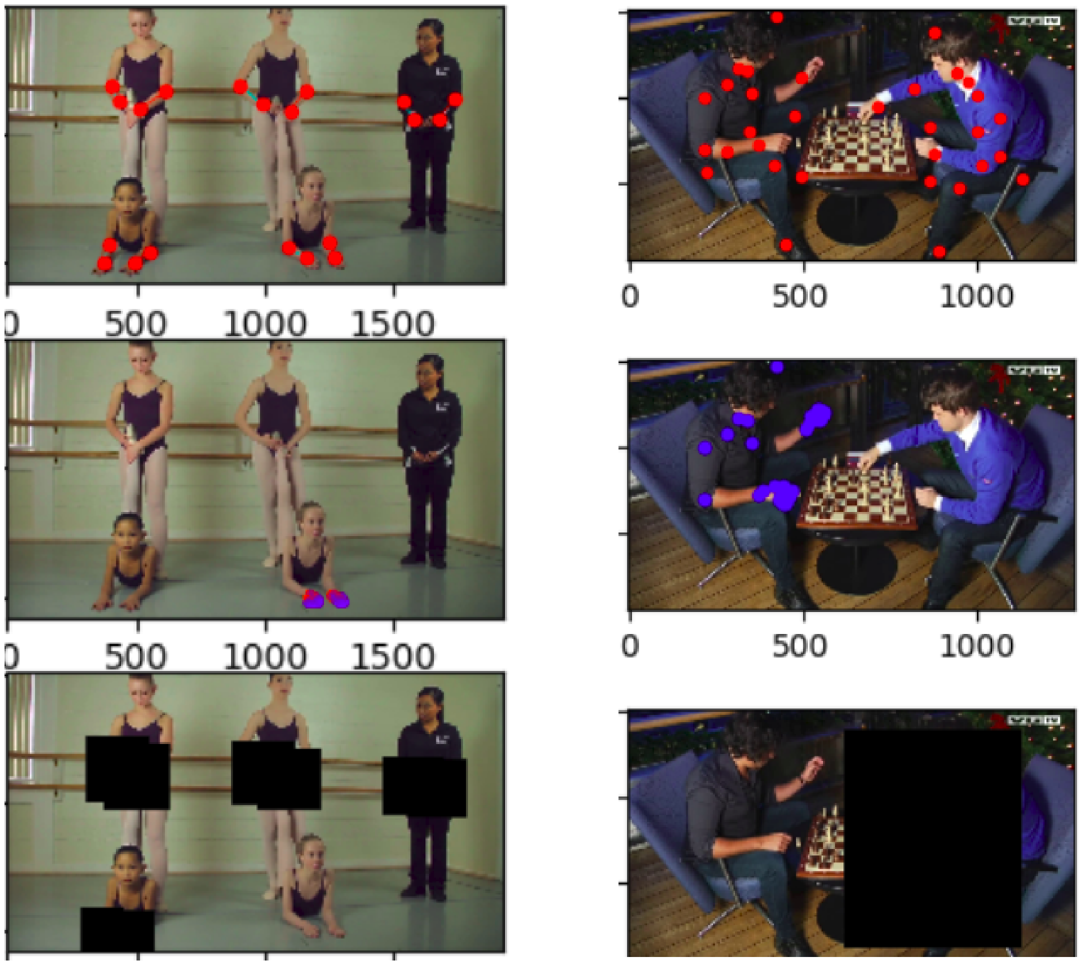

FIGURE 5.1: Left images: privileged masking for the hand model. Right images: privileged masking for the Hand-body-nzl model. Top row: available keypoints of the original MPII dataset. Middle row: available keypoints of the MPII+NZL dataset. Bottom row: black bounding boxes representing the masked pixels around the instances without annotations.

- **Data fusion**: There are only a few public dataset with hand annotations to train on, and what is even worst, each of that datasets use to have different annotation methodologies (e.g. different number of keypoints or in different locations). Hence, the data fusion allow to train with multiple datasets with different annotation methodologies. This preprocessing strategy have been inspired by the MPII+NZL and the Panoptic datasets, where the first provides annotations for the body and both hands and the second only provides the right hands annotations. To deal with that issue, first the privileged masking had been applied to the MPII+NZL dataset, masking all hands without annotations. Then, we have performed an inverse masking in each image of the Panoptic dataset. Due that only the right hands are annotated in that dataset, we have used a bounding box containing each right hand, and have masked the rest of the pixels in the image, and our network only will be penalized if an error inside the bounding boxes around the hands is made, leaving the rest of the image without penalization during training. Then, both preprocessed datasets are shuffled together.

During the development of the OpenPifPaf4hands, we noticed that the method presented high sensibility to the initial size of the input images. This was due that the original method was tuned specifically for the COCO dataset, where all images have the same size. Hence,

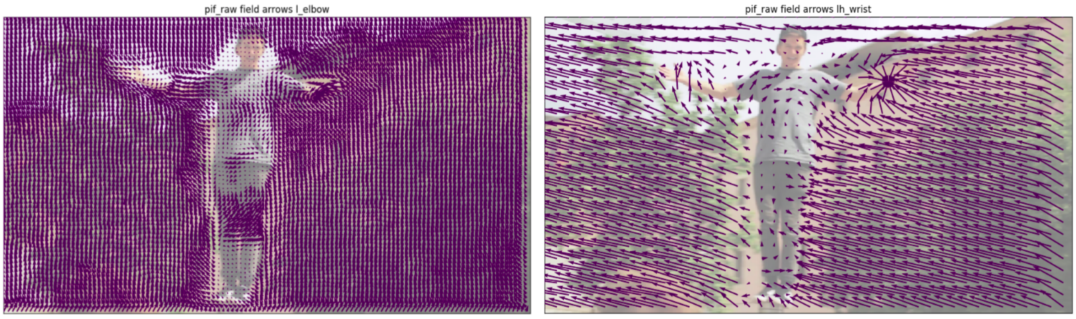

FIGURE 5.2: Visualization of the paf vectors during inference. In the left, the output paf vector fields for the left elbow with an image which rescaling at the preprocessing stage provided retained too much resolution. In the right, the output paf vector fields for the wrist with a proper rescaling at the preprocessing stage, where it could be clearly seen that only in the left wrist all paf vectors point to the same point.

		Hand	Hand-body-nzl	Hand-retrained
MPII-NZL	**Seed-threshold**	0.01	0.01	
	Instance-threshold	0.01	0.01	
Panoptic	**Seed-threshold**	0.01		0.001
	Instance-threshold	0.01		0.1

TABLE 5.1: Threshold values for the different models and datasets.

some preprocessing image transformations like the image rescaling were carried relative to the input image size and the final resolution was not adequate to the method. To face the reduced number of predictions that the method was providing for the MPII+NZL dataset, we coded a visualization module to compare between all the *pif* and *paf* head networks predictions and the targets just before the loss computation (more details about the *pif* and *paf* head networks fields could be checked in [2]). From that visualizations, we achieved to set the rescaling image transformation to achieve a satisfactory resolution in the image after preprocessing and the method accuracy hugely improve its accuracy. In the appendices section B.1, some images obtained during the described size tuning are shown.

After tuning the training, the difference in the results could be seen in the figure 5.2, in which could be noticed that while in the left image the paf vectors are concentrated around multiple parts of the image, in the right image only in the left wrist all paf vectors are pointing to the same direction, and the rest of vectors present an isotropic distribution.

5.5 Decoder parameters tuning

Once the training stage had been properly tuned, still the decoder parameters needed to be changed to adapt to our hand model. Among these parameters, the instance and seed thresholds should be carefully selected. After a grid search, we found the top and bottom limits between we could obtain satisfactory results. Even thought, if the bottom limit was used, our method could not properly handle the large number of predictions and overlapped predictions were obtained. In the other hand, if the top limit was used, only a few poses of all the hands in the images were detected. Finally a trade-off between both limits was used, setting the threshold values presented in the table 5.1. The difference of the predictions using different threshold values could be appreciated in figure B.5.

FIGURE 5.3: OpenPifPaf4hands predictions of the MPII dataset using the hand model.

5.6 OpenPifPaf4hands evaluation

Even that we have not still reached the 3D pose prediction with the OpenPifPaf4hands, we think that it represent a positive contribution to the 2D hand pose detection. Hence, we have evaluated OpenPifPaf4hands in the MPII+NZL and the Panoptic datasets, making a comparison of the results with OpenPose, for being one of the most similar approaches. Some qualitative results of our method in the MPII+NZL and Panoptic datasets with the hand model are presented in figures 5.3 and 5.5, while in 5.4 the results with the hand-body-nzl model are shown. In the figure 5.5, also an inaccurate prediction.

Despite this satisfactory results, there are images with hands in which our method is not able to detect any pose. The most challenging dataset is the MPII subset of the MPII+NZL dataset due to its small hands, high occlusion, high number of humans in each image and blurry hands.

For the quantitative evaluation, we have chosen the PCK, as the one used by OpenPose authors in [10]. The PCK tells what percentage of keypoints are predicted below a threshold error. We will present two different PCK plots, the one with normalized distance errors (OpenPose only provide this results in their work) and the standard distance errors PCK. To

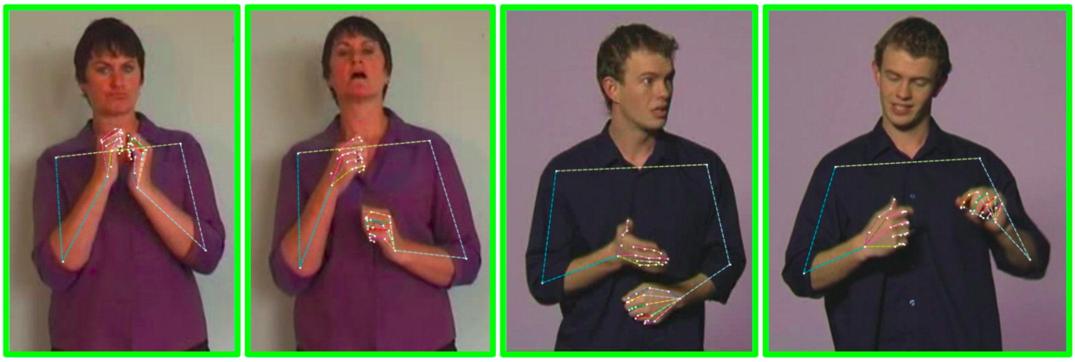

FIGURE 5.4: OpenPifPaf4hands predictions of the NZL dataset using the hand-body-nzl model.

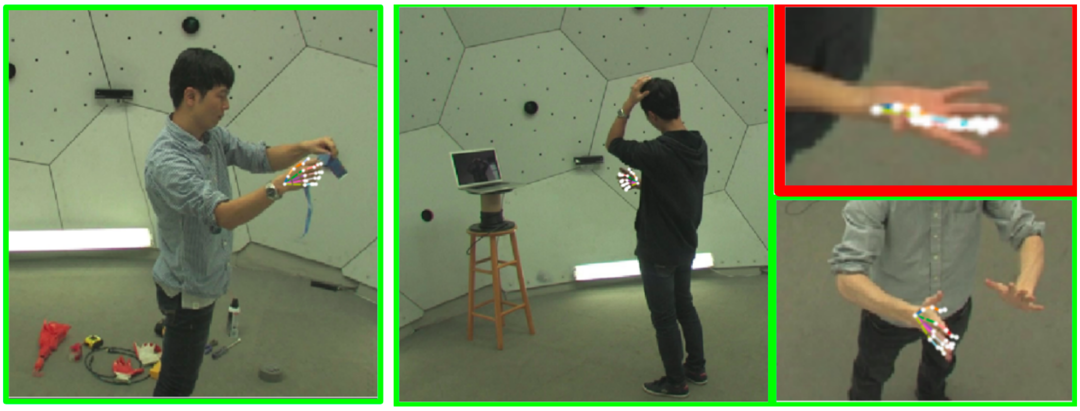

FIGURE 5.5: OpenPifPaf4hands predictions of the Panoptic dataset using the hand model. With a red frame, an unsatisfactory estimated hand pose is shown, in which only the palm keypoint is close to the ground truth. It should be noted that due that on the Panoptic dataset only the annotations of the right hands are provided, we only predict the hand pose of the right hands.

normalize the errors, for each hand its minimum bounding box its computed and the maximum value of length or wide of the bounding box is taken as the reference distance. Then, each error is divided by this reference distance, clipping the maximum error to 1. The normalized distance error is more fair than the standard because it considers the same error in a big hand at the front of the image and a little hand in the back. Additionally and to understand more the differences between each method and model, we also present the errors in pixels without any clipping.

The results of the hand pose prediction presented in [10] could not be used directly because at the moment of the publication, they was not still using the body pose to crop the hand images but they use a hand detector. It was not until the publication of [8] that body and hand pose were tied together. Therefore, we have reproduced their results thanks to that they provide their code for inference. At the moment, they provide two body pose estimators, one trained in the COCO dataset and another trained on the MPII dataset, we have reproduced the results with both to compare them to our method. For the sake of simplicity, from now on we will refer the OpenPose model trained with the COCO dataset as OpenPoseCOCO and the one trained with the MPII as OpenPoseMPII.

The results of our method with the hand-body-nzl and hand models and the OpenPose method in the images in which there is at least one keypoint detected are plotted in the figure 5.6. The normalized error is clipped to a maximum of 1 so, if one keypoint has been predicted out of a hand bounding box, it's error will be 1. This fact explains the increase of keypoints number when in the x axis the normalized distance error is 1.

As could be seen, the model that performs the best in the MPII+NZL dataset is OpenPoseCOCO. Nevertheless, all of our three models outperform OpenPoseMPII. If we take into account that our hand and hand-body-nzl models have only been trained in the MPII+NZL dataset, which is a small subset of the MPII datset and the NZL dataset, which is itself small, we think that the results that we have obtained are satisfactory.

It is important to notice that when the methods are tested in the MPII and the NZL subsets of the MPII+NZL separately, the root of the lack of performance of OpenPoseMPII dataset arises, pointing out that because their method first need to detect all hands using the body pose prediction, and their body pose predictor has only been trained on the MPII dataset, the hand poses detected in the NZL subset have much larger errors than in the MPII subset. The accuracy obtained by our hand-body-nzl model in the MPII subset is similar than OpenPoseMPII, while our hand model is able to outperform it, specially if normalized errors greater than 0.3 are allowed. In the case of the NZL, in contrast than in the MPII subdadaset, our hand-body-nzl model performed better than our hand model, and both achieved better accuracy than OpenPoseMPII. In all cases, OpenPoseCOCO dataset outperform all of our methods, and our model trained using data fusion of the MPII+NZL and Panoptic datasets at the same time produces always worst results than our model trained only in the MPII+NZL dataset, which point out that the inverse masking strategy with the Panoptic dataset is not enough to fuse both datasets.

Additionally to the evaluation indicators of [10], we also have computed the distance in pixels from each predicted keypoint to its ground truth (figure 5.7). This indicator provides information about the performance of the methods in terms of errors without normalization. The PCK with the normalized distance attribute the same error to the predictions made out of the bounding box of the hands, making no difference between a prediction that has been made just at the bounding box border or at the other side of the image. The error without normalization take into account that difference and provide additional information to compare the studied methods. Using this indicator, both of our models outperform OpenPose in the images in which there is at least one keypoint detected. Because in this case there is no normalization nor maximum error clipping, the hands at the front, hence, occupying more pixels, produce larger errors in their predictions than the little hands at the back, occupying few pixels. Then, our conclusion is that the smaller the hands, the worst OpenPose perform in comparison to our method. The results when only the NZL subdataset is used give even more sense to our intuition in the case of OpenPoseMPII, due that the hands in this subset are clearly bigger and at the foreground of the image in comparison with the ones in the MPII subset.

Due that in the Panoptic dataset only one hand is annotated, we only have used our hand model in it. In figure 5.8, as expected, the huge error difference between our method trained on the MPII+NZL dataset and our method trained on the Panoptic dataset could be seen. When the distance error is normalized for the PCK computation, both OpenPose methods are the ones that perform the best. Even though they are also predicting more keypoints out of the hands bounding boxes than OpenPifPaf4hands trained on Panoptic. The reason of the best accuracy of OpenPose in this dataset is because their hand pose predictor has been trained in that dataset, and not only in the provided images but also from all the 31 simultaneous viewpoints that they had in the room where the dataset was recorded, therefore, when an image crop is accurate, their hand pose predictor should perform quite well because it approaches the hands one by one. When the error is not normalized by the hand size, our methods perform better, even if the method has not trained in the Panoptic dataset, pointing out again that our method perform better for the hands in the foreground.

Until now, only the predictions in images with at least one keypoint predicted have been evaluated. Therefore, due that neither in the MPII+NZL nor in the Panoptic dataset there are images without any annotation, the number of images without any detection in relation with the total number of images could be used as a performance indicator. This data is presented in figure 5.9 and 5.10.

Here is where the pitfall of our algorithm at this stage arises, the OpenPifPaf4hands suffer from not predicting any keypoint in a considerable quantity of images in the MPII subdataset. It is interesting to note that the number of images without predictions by our hand-body-nzl model is specially high for the MPII dataset, in which many times some of the keypoints of the hand-body-nzl model are occluded. This points out that we have to put our focus in the instance model reconstruction from the keypoints detected by the pif headnet, because it seems that if there is one non-detected keypoint of the middle of the instance model connections, OpenPifPaf4hands don't build the rest of the connections. In this case, both OpenPose models predicted keypoints in more images than our models.

In figures 5.11 and 5.12, the computation times of all the images in the MPII+NZL dataset and subdatasets and the Panoptic dataset are presented. During inference time, all methods have used 2 GPUs GeForce GTX 1080 Ti. Despite we thought that our method will be faster than OpenPose, this graph throws the evidence that this is not true, but this is probably due that OpenPose libraries are compiled in C++, which is usually much faster than Python. Surprisingly, the only method that spend the same time per image in average is our hand model with fusion data preprocessing in the case of the MPII+NZL dataset.

It also should be noticed that because of the MPII lighter body model (15 keypoints instead of 18), the OpenPoseMPII always have been considerably faster than the OpenPoseCOCO model.

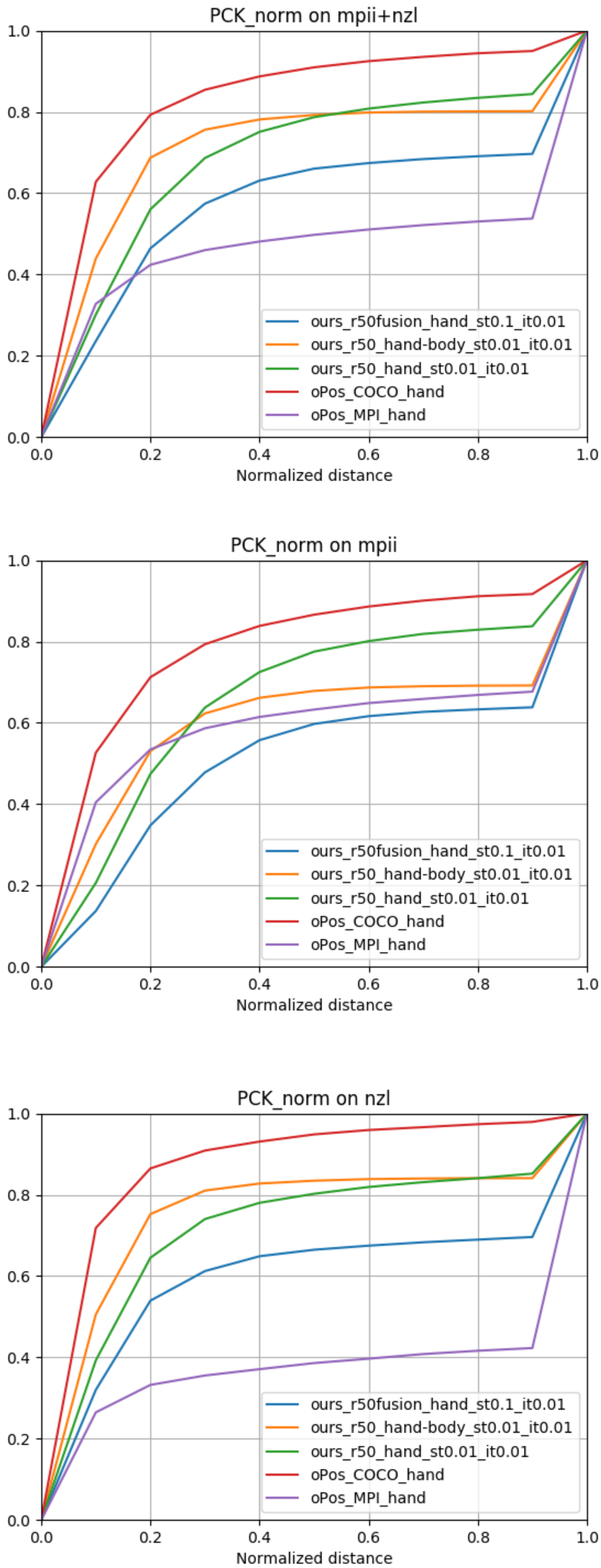

FIGURE 5.6: PCK plots with distance normalized for MPII+NZL dataset, only the MPII and only the NZL subdatasets. Only the images where at least one keypoint has been predicted has been included in the plot data.

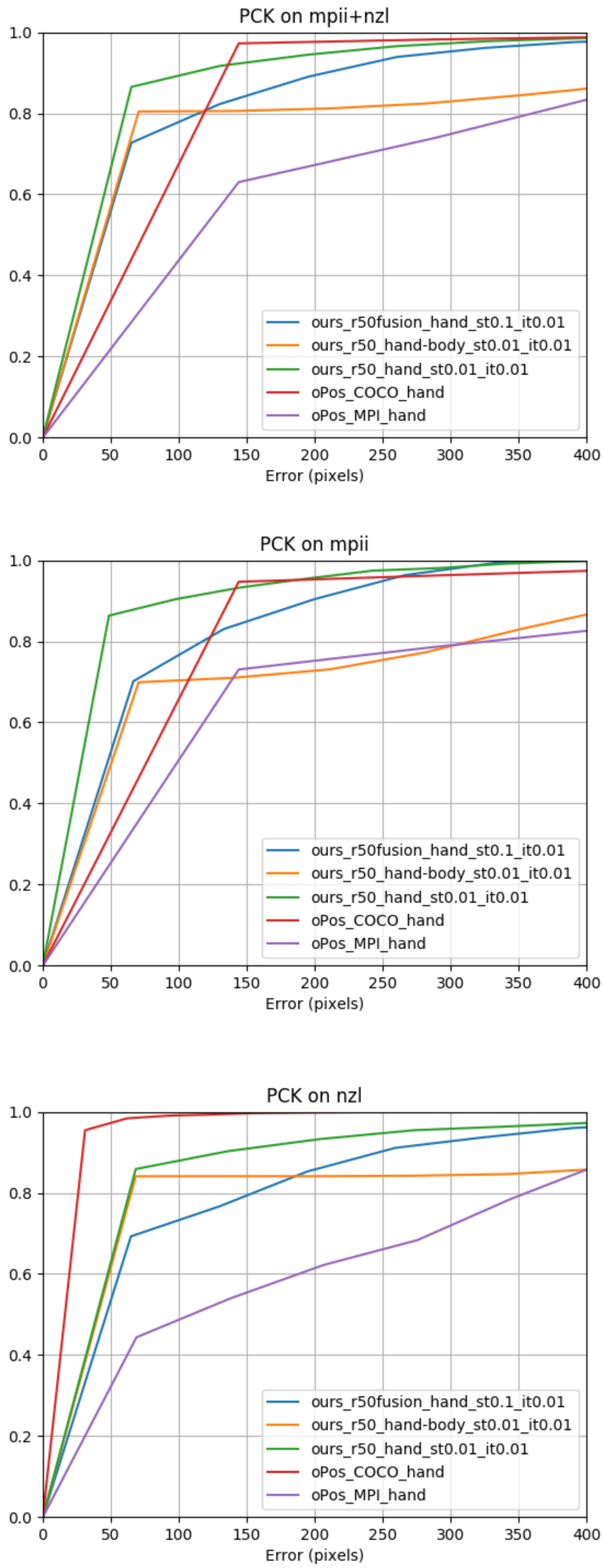

FIGURE 5.7: PCK plots with the error in pixels distance from the ground truth to the predicted keypoints for MPII+NZL dataset, only the MPII and only the NZL subdatasets. Only the images where at least one keypoint has been predicted has been included in the plot data.

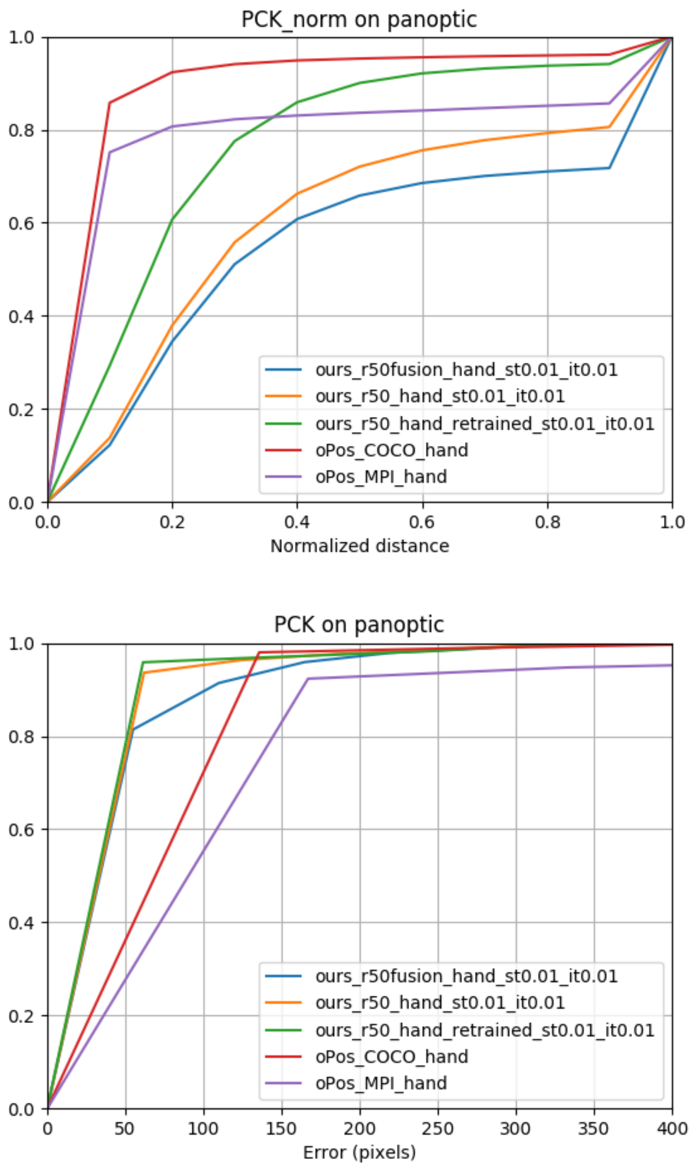

FIGURE 5.8: PCK plots with distance normalized and without normalization for the Panoptic dataset.

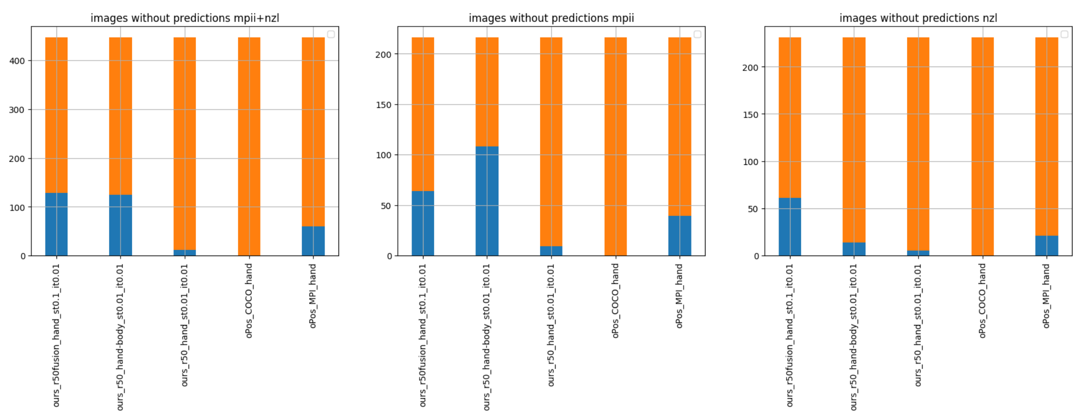

FIGURE 5.9: MPII+NZL dataset. Orange: total number of images in the dataset. Blue: total number of images without any keypoint predicted.

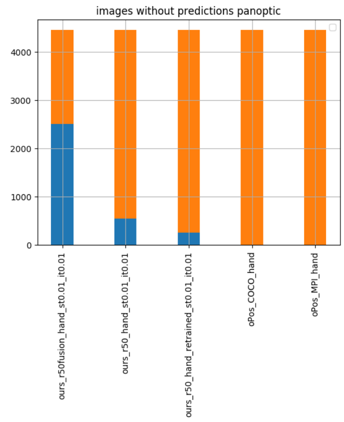

FIGURE 5.10: Panoptic dataset. Orange: total number of images in the dataset. Blue: total number of images without any keypoint predicted.

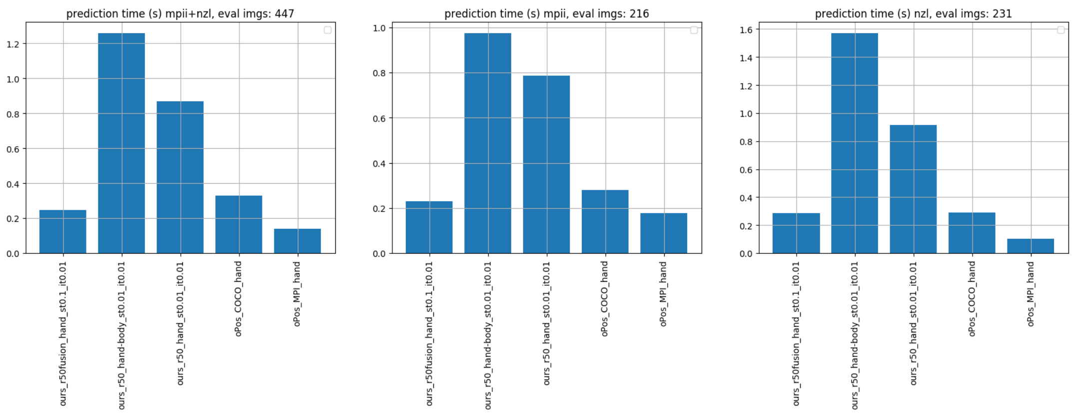

FIGURE 5.11: Prediction times for the MPII+NZL dataset and it's sub-datasets.

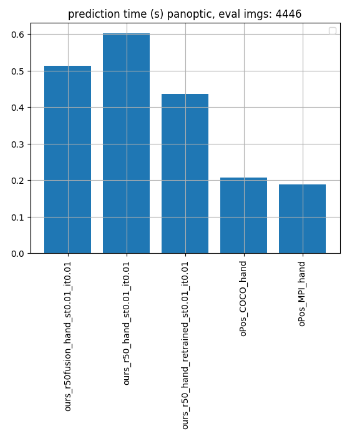

FIGURE 5.12: Prediction time for the Panoptic dataset.

Chapter 6

Experiments

We have evaluated our finger width measurement pipeline in two different situations. The first is by means of our own dataset, and the second is the evaluation of the method running in real-time. Because ring sizes change one millimeter from one size to the other, we set the maximum error of $1mm$ for a finger width prediction to be accepted. In the figure 6.1, some finger-width predictions using images of our dataset are presented. Among the images in the figure 6.1, two of the images are inside our acceptance rate and the other two are typical failure cases, which include a lack of illumination homogeneity or the lack of separation between the fingers, what make the method to consider two fingers as just one.

The prediction errors of all the images in our dataset are presented in the table 6.1. At the end of each step of the pipeline, some conditions should be fulfilled, otherwise the sample is discarded. While testing the method, it has been noticed that the condition to fulfill with more impact in the results is the allowed range of values of the eigenvectors in the OBB step of the preprocessing 4.4.1. Therefore, we have evaluated our method with two different sets of restrictions for these eigenvectors, one more strict and another with relaxed restrictions. As could be seen in the presence columns of the table 6.1, if the constraints are relaxed the number of valid measurements is much numerous, or what is the same, the number of discarded samples is scarcer. Therefore, this strategy lead to a faster final measurement at the expenses of having less accuracy. The obtained average error is $3.87mm$ and $2.04mm$ for the left and right hand respectively.

In the other hand, if the strict constraints are used, the average error decrease to $1.65mm$ and $0.28mm$ for the left and right hand respectively, which means that the target of less than $1mm$ error will be fulfilled for right hands and we would be very close for the left hands. The inconvenience of this strategy is that it makes the method much slower, and even in some cases, any of the dataset images have considered as valid for some subjects.

	results with relaxed constraints				results with strict constraints			
	left hand		right hand		left hand		right hand	
Subject id	**presence**	**error**	**presence**	**error**	**presence**	**error**	**presence**	**error**
1	0		6	2.07	0		0	
2	40	5.67	0		1	2.66	0	
3	44	18.99	2	2.19	2	1.425	0	
4	64	0.38	8	1.56	2	0.17	1	0.04
5	18	3.03	4	3.60	1	0.41	0	
6	18	1.89	5	2.33	0		0	
7	57	0.29	0		3	1.68	0	
8	38	3.84	0		0		0	
9	17	0.44	0		1	1.27	0	
10	24	2.40	9	2.05	2	1.96	1	0.51
11	12	3.37	0		1	2.04	0	
12	47	1.91	0		3	1.32	0	
13	64	4.44	1	0.48	2	3.56	0	
14	2	3.69	0		0		0	
MAX	64	18.99	9	3.60	3	3.56	1	0.51
MIN	0	0.29	0	0.48	0	0.17	0	0.04
AVG		3.87		2.04		1.65		0.28
SUM	445		35		18		2	

TABLE 6.1: Average errors in mm of the finger width predictions for each hand of each of the 14 subjects of our dataset. The left half present the results with relaxed constraints and the right half the results with strict constraints The presence columns indicate the number of instances of that hand and subject have been considered valid for the measurement. In the bottom 4 rows, the maximum, minimum, average and sum of all the errors for each column are presented.

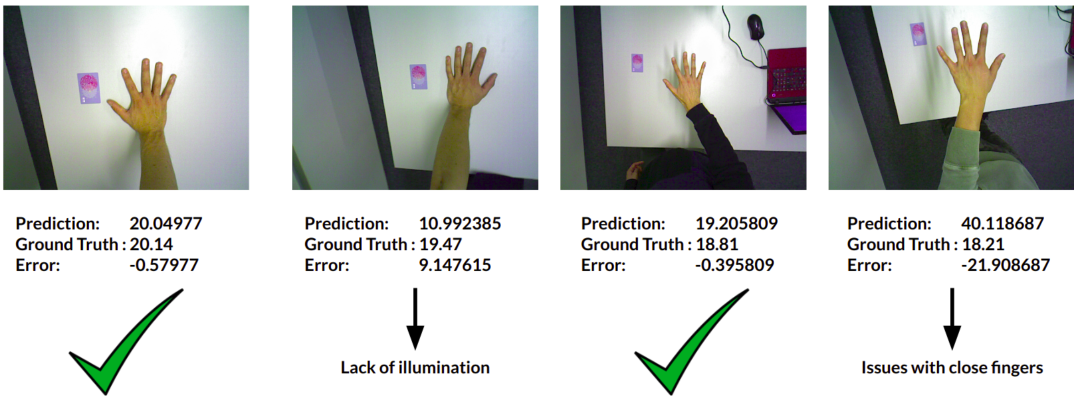

FIGURE 6.1: Examples of positive and negative finger-width predictions.

Chapter 7

Conclusions

A pipeline able to measure the ring finger width in real-time has been implemented, to our knowledge, there is no other public work that has been done the same. Moreover, the error has demonstrated to be of less than $1mm$ under constrained conditions, allowing it to estimate the ring size quite accurately. Even if in this project the 3D reconstruction has been oriented to the ring finger width measurement, just the 2 last steps of our pipeline have to be modified in order to easily perform other measurements. What is more, the target instance could be different than a hand, and therefore, instead of using a hand pose predictor, other methods could be used to obtain context information about the target and following perform the measurement of the desired dimension.

To evaluate our method, a new dataset with both hands of 14 different subjects has been recorded from different viewpoints, this dataset could be used to evaluate 3D hand reconstruction or hand pose methods in future works.

A new method bottom-up multi-person for hand-body pose detection evolved from OpenPifPaf, has been presented, proving it's ability to deal with small, low resolution, occluded and with a wide range of different hand poses. To our knowledge, this will be the first public method of this kind. From the comparison between OpenPifPaf4hands and OpenPose, it has been noticed that the training in the COCO dataset suppose a huge advantage for body pose estimation in comparison with the original MPII dataset. This remarks that COCO dataset has much more examples and provide enough variability to allow transfer learning, even making possible to achieve better predictions in the MPII dataset than the same network trained on the self MPII dataset. Therefore, our aim is to prepare our data fusion preprocessing to be able to train with multiple datasets at the same time, including COCO among them.

Chapter 8

Future work

The next step would be to train the OpenPifPaf4hands with multiple datasets at the same time using the fusion data preprocessing. Make the 3D hand pose predictions from the 2D current predictions of the OpenPifPaf4hands and substitute the hand detection and hand pose predictor steps in the finger width measurement pipeline. Regarding the finger width measurement pipeline, a deep learning algorithm could be trained to perform the width measurement, avoiding the time consuming step of the region growing. The robustness against occlusion and viewpoints different than the perpendicular view should be improved in order to make the algorithm more reliable.

Appendix A

Finger width measurement

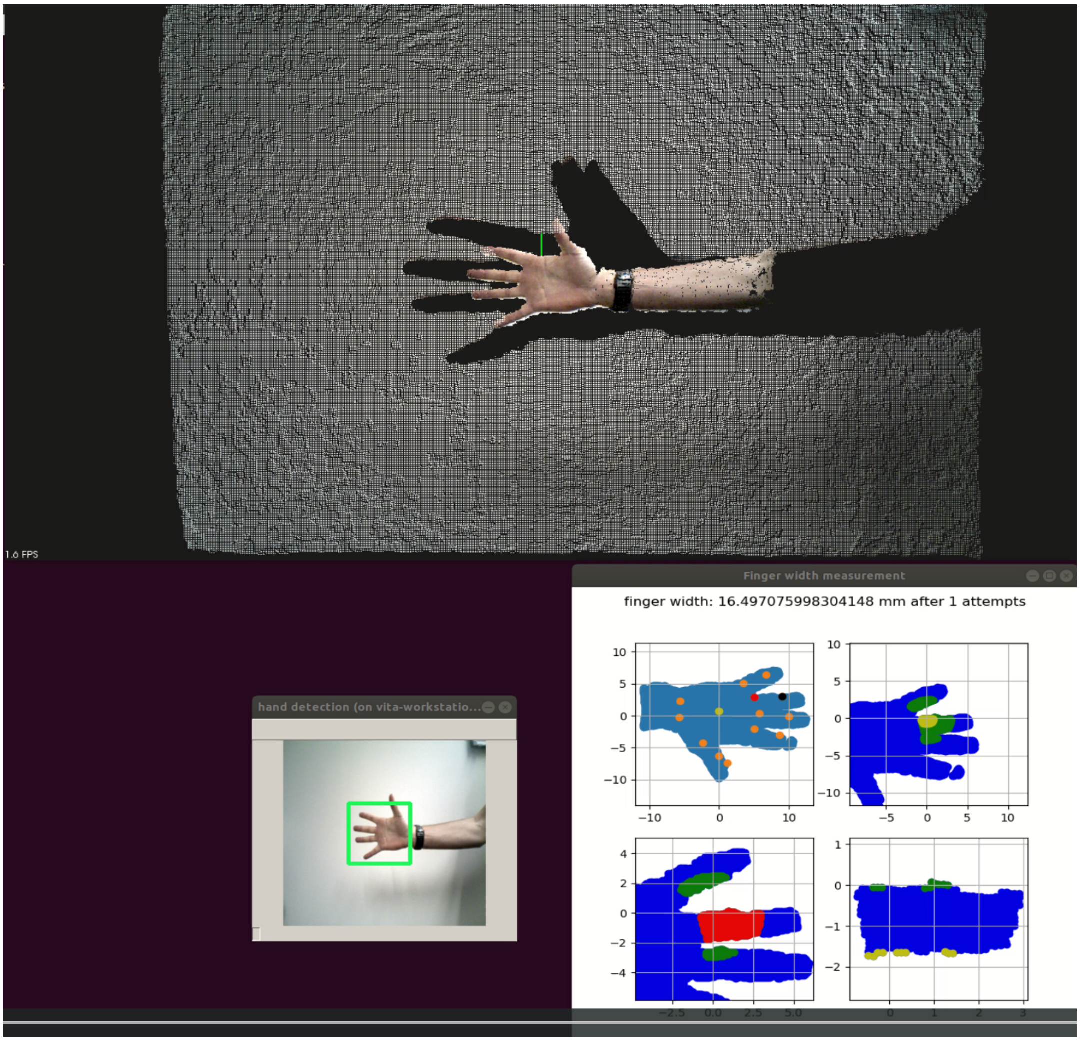

FIGURE A.1: Capture of the graphical interface of the finger measurement working in real-time. The window above show the point cloud, in the bottom left window the RGB image extracted from the point cloud and the bounding box estimated by the hand detector could be seen. In the bottom right window, the steps of the finger measurement explained in section 4.5 could be seen.

Appendix B

OpenPifPaf4hands

B.1 Training loss visualization

In this section, some of the visualizations during training with the MPII dataset are presented.

It should be noticed that only the left hand palms of the human at the left are learned in figures B.1 and B.2, while the network is learning that for each of the other two humans in the image, the left hand palms could be located almost anywhere in their body, this is due that this images where produced using privileged masking training (section 5.1) and due that only the hand annotations of the left human were provided, the entirely bodies of the other two humans were masked during training.

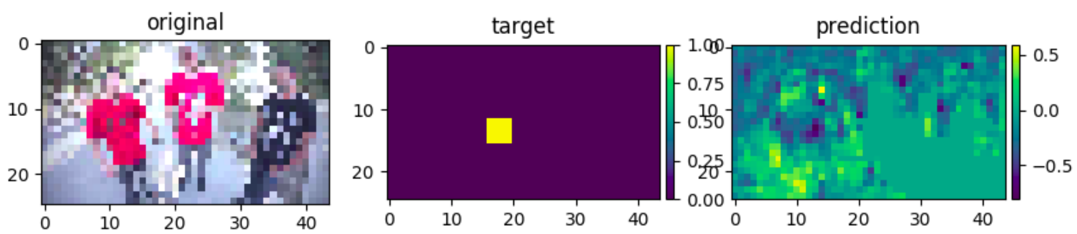

FIGURE B.1: Target and prediction at epoch 0 of training for pif intensity of the left hand palm.

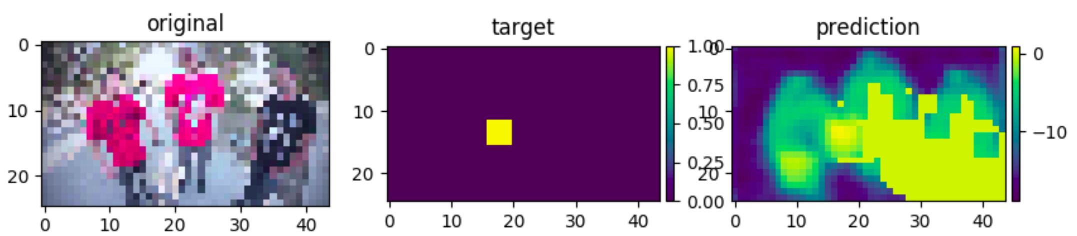

FIGURE B.2: Target and prediction at epoch 150 of training for pif intensity of the left hand palm.

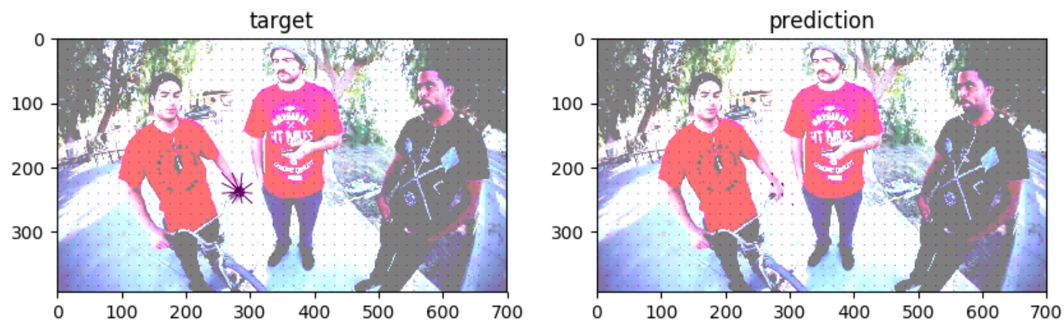

FIGURE B.3: Target and prediction at epoch 0 of training for paf vectors pointing to the left hand palm.

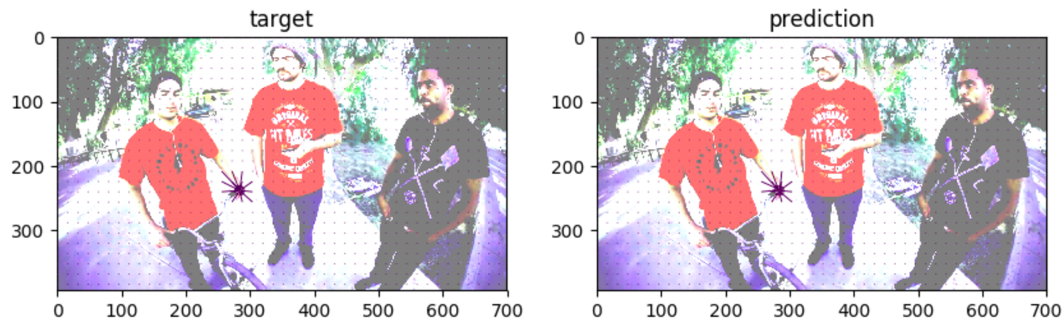

FIGURE B.4: Target and prediction at epoch 90 of training for paf vectors pointing to the left hand palm.

B.2 Prediction tuning: decoder parameters setting

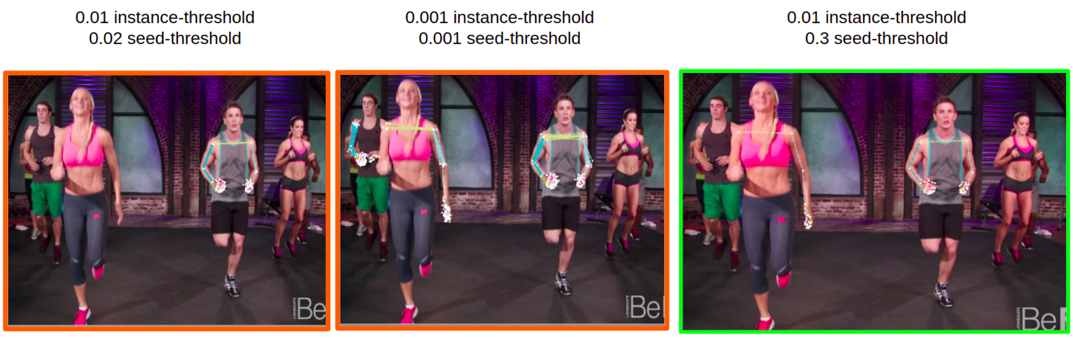

FIGURE B.5: Predictions for the hand-body-nzl model with different threshold values in MPII dataset crowded images.

Bibliography

[1] T. G.N. F. Neverova N. Wolf C., “Hand segmentation with structured convolutional learning. in: Cremers d., reid i., saito h., yang mh. (eds) computer vision – accv 2014. accv 2014. lecture notes in computer science, vol 9005. springer, cham”, 2015.

[2] S. Kreiss, L. Bertoni, and A. Alahi, “Pifpaf: Composite fields for human pose estimation”, *CVPR, arXiv preprint arXiv:1903.06593*, 2019.

[3] W. Garage, *The point cloud library (pcl)*, Last accessed 12 April 2019, 2019. [Online]. Available: `http://pointclouds.org/`.

[4] D. Victor, *Real-time hand tracking using ssd on tensorflow*, `https://github.com/victordibia/handtracking`, 2017.

[5] S. Bambach, S. Lee, D. J. Crandall, and C. Yu, “Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions”, in *The IEEE International Conference on Computer Vision (ICCV)*, 2015. [Online]. Available: `http://vision.soic.indiana.edu/projects/egohands/`.

[6] J. W.J. Y. Liuhao Ge Yujun Cai, *Hand pointnet: 3d hand pose estimation using point sets. in: Proc. ieee conf. comput. vis. pattern recog. pp. 8417–8426*, `https://sites.google.com/site/geliuhaontu/home/cvpr2018`, Last accessed 17 April 2019, 2018.

[7] A. A. Paschalis Panteleris Iason Oikonomidis, “Using a single rgb frame for real time 3d hand pose estimation in the wild”, 2017.

[8] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, “OpenPose: Realtime multi-person 2D pose estimation using Part Affinity Fields”, in *ArXiv preprint arXiv:1812.08008*, 2018.

[9] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, “Realtime multi-person 2d pose estimation using part affinity fields”, in *CVPR*, 2017.

[10] T. Simon, H. Joo, I. Matthews, and Y. Sheikh, *Hand keypoint detection in single images using multiview bootstrapping (cvpr).* `https://www.cs.cmu.edu/~tsimon/projects/mvbs.html`, 2017.

[11] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, “Convolutional pose machines”, in *CVPR*, 2016.

[12] G. G.-H.T.-K. K. Shanxin Yuan Qi Ye, “The 2017 hands in the million challenge on 3d hand pose estimation”, 2017.

[13] B. S.G.M.J.Y.C.K.M.L.P.M.J.K.S.H.L.G.J.Y.X.C.G.W.F.Y.K.A.Y.W.Q.W.M.M.S.E.S.L.D.L.I.O.A.A.T.-K. K. Shanxin Yuan Guillermo Garcia-Hernando, “Depth-based 3d hand pose estimation: From current achievements to future goals”, 2018.

[14] S. L.X.T.J. S. Xiao Sun Yichen Wei, “Cascaded hand pose regression”, 2015.

[15] J. Tompson, M. Stein, Y. Lecun, and K. Perlin, “Real-time continuous pose recovery of human hands using convolutional networks”, *ACM Transactions on Graphics*, vol. 33, 2014.

[16] F. Mueller, D. Mehta, O. Sotnychenko, S. Sridhar, D. Casas, and C. Theobalt, "Real-time hand tracking under occlusion from an egocentric rgb-d sensor", in *Proceedings of International Conference on Computer Vision (ICCV)*, 2017. [Online]. Available: https://handtracker.mpi-inf.mpg.de/projects/OccludedHands/.

[17] C. Zimmermann and T. Brox, "Learning to estimate 3d hand pose from single rgb images", in *IEEE International Conference on Computer Vision (ICCV)*, https://arxiv.org/abs/1705.01389, 2017. [Online]. Available: https://lmb.informatik.uni-freiburg.de/projects/hand3d/.

[18] S. L. Christopher Ham and S. Singh, "Hand waving away scale", in *ECCV*, 2014.

[19] Y. Z.P.T.L.C.H. W. Xiaoming Deng Shuo Yang, "Hand3d: Hand pose estimation using 3d neural network", in *CVPR*, 2017.

[20] F. Mueller, F. Bernard, O. Sotnychenko, D. Mehta, S. Sridhar, D. Casas, and C. Theobalt, *Ganerated hands for real-time 3d hand tracking from monocular rgb*, https://handtracker.mpi-inf.mpg.de/projects/GANeratedHands/, 2018.

[21] M. O. Mahdi Rad and V. Lepetit, "Feature mapping for learning fast and accurate 3d pose inference from synthetic images", in *CVPR*, 2018.

[22] F. N.K.V.K.T.A.H.D. S. Jameel Malik Ahmed Elhayek, "Deephps: End-to-end estimation of 3d hand pose and shape by learning from synthetic depth", in *CVPR*, 2018.

[23] ASUS, *Asus xtion pro*, https://www.asus.com/3D-Sensor/Xtion_PRO/, Last accessed 12 April 2019.

[24] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele, "2d human pose estimation: New benchmark and state of the art analysis", in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[25] S. B.L.B.R.G.J.H.P.P.D.R.C.L.Z.P. D. Tsung-Yi Lin Michael Maire, "Microsoft coco: Common objects in context", in *CVPR*, 2015.

[26] C. P. Yangang Wang Member IEEE and Y. Liu, "Mask-pose cascaded cnn for 2d hand pose estimation from single color image", in *IEEE Transactions on Circuits and Systems for Video Technology PP(99)*, 2018.